IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Applicants: Gschwind, Michael K. Examiner: Choi, Woo H.

Serial No: 09/940,911 Group Art Unit: 2189

Filed: August 28, 2001 Docket: 8728-545 (YOR9-2001-0606)

For: **METHOD AND APPARATUS FOR ALIGNING MEMORY WRITE DATA IN A MICROPROCESSOR**

Mail Stop Appeal Brief
Commissioner of Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

In response to the Final Office Action dated December 23, 2004, rejecting Claims 1-7, 11-13, 17-20, and 22-24 under 35 U.S.C. §102, and of Claims 8-10, 14-16, and 21 under 35 U.S.C. §103, Applicant appeals pursuant to the Notice of Appeal dated April 25, 2005, and submits this appeal brief.

CERTIFICATE OF MAILING (37 C.F.R. § 1.8a)

I hereby certify that this paper (along with any paper referred to as being attached or enclosed) is being deposited with the United States Postal Service on the date shown below with sufficient postage as first class mail in an envelope addressed to the: Mail Stop Appeal Brief, Commissioner of Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on **July 25, 2005**.

Date: 7/25/05 David L. Heath

David L. Heath

08/15/2005 EFLORES 00000071 500510 09940911
01 FC:1402 500.00 DA

TABLE OF CONTENTS

1. Real Party in Interest

The real party in interest is International Business Machines Corporation of Armonk, New York, the assignee of Michael K. Gschwind, Martin E. Hopkins, and H. Peter Hofstee, the inventors of the application.

2. Related Appeals and Interferences

None.

3. Status of Claims

Claims 1-24 are pending and stand rejected, claims 25-29 having been canceled has being directed to an unelected invention. Claims 1 to 24 and are under appeal. A copy of the claims as pending is presented in the Appendix.

4. Status of Amendments

No after final Amendments were filed in this case.

5. Summary of Claimed Subject Matter

The present invention provides a system and method that supports a software based alignment of memory accesses as well as big-endian and little-endian data formats (pg. 5, l. 3-7). In a big-endian format, the higher order data unit is located at the lower numbered unit address, while in a little-endian format, the higher order data unit is located at the higherr numbered unit address (pg. 2, l. 8-12).

In one embodiment of the invention that uses an alignment network that does not support subword alignment, a sub-width data item is rotated into an appropriate slot of a full width data item, and is then stored into memory using a byte-wise write-enable functionality (pg. 23, l. 1-9). In one embodiment of the invention, this rotate and store can be implemented by a three instruction sequence as illustrated in FIG. 7 (pg 23, 20-21). The rotate instruction rotates a register (r4) left by the byte count specified by the 2 rightmost bits of another register (r3), and

the result is stored in a third register (r2) of the preferred data width. This third register contains a correctly aligned data item. (Pg. 24, l. 6-9.) The store instruction first computes a write mask from the from an address agrument for the specified width. The address argument can take to form of a base address and an offset (pg. 25, l. 1-5). The store instruction then supplies the result of the rotate instruction to the memory interface under control of the computed write mask (pg. 26, l. 3-7). For the purposes of the present invention, the terms "mask" and "write mask" refer to a control word that selects data bits, bytes, or words from a collection of first, second, etc., data collections of bits, bytes, or words, selecting for each resultant bit, byte, or word, a bit, byte, or word from either the respective value in the first, second, etc., data collections under control of the control word. The mask is formed from the address argument suppiled based on the store datatype to facilitate the writing of a number of bits corresponding to the size of the data type. Thus, if the mask encodes bits, then the number of bits set in the mask will be set to the number bits in the data type in a common encoding of masks. (Pg. 25, l. 5-20.)

In another embodiment of the invention that uses a processor containing wide vector registers, 32-bit words can be stored from a right aligned wide vector register. In one embodiment of the invention, this instruction sequence can be implemented by a three instruction sequence as illustrated in FIG. 10. (Pg. 27, l. 3-8.) The vrotate instruction rotates a register (r4) left by the byte count specified by the 4 rightmost bits of another register (r3), and the result is stored in a third register (r2) of the preferred data width (pg. 27, l. 22 – pg. 28, l. 2). This third registrer contains a correctly aligned data item. The vstore instruction first computes a write mask from the from an address agrument for the specified width. The address argument can take to form of a base address and an offset. The vstore instruction then supplies the result of the vrotate instruction to the memory interface under control of the computed write mask. (Pg. 28, l. 8-15.)

Independent claims 1, 12, and 19 are representative of the claimed subject matter that embodies features of the invention as generally described above. For illustrative purposes, the cliamed subject matter will be described with reference to exemplary embodiments described in Applicant's specification, and accompanying figures, although nothing herein should be construed as unduly limiting the scope of the claimed subject matter. For each claim listed

below, the claim elements are presented in italized text, and are followied by a citation to exemplary figures and/or supporting text in the specification.

Claim 1:

A method for aligning and inserting data elements into a first memory based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction, (see FIG. 7, pg. 23-26, FIG. 10, pg. 26-28) *comprising the steps of:*

given a data item that includes a data element to be stored,

aligning the data element in a second memory with respect to a predetermined position in the first memory, in response to the one or more alignment instructions; (FIG. 8, pg. 24, l. 1-9, FIG. 11, pg. 27, l. 20 – pg. 28, l. 2)

dynamically generating a mask to enable writing of memory bit lines that correspond to the aligned data element;(FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3) *and*

writing the memory bit lines to the first memory under a control of the mask, wherein said generating and writing steps are performed in response to the single store instruction. (FIG. 9, step 920, pg. 26, l. 3-6, FIG. 12, step 1220, pg. 28, l. 12-15).

Claim 3:

The method of claim 1, further comprising the step of computing the mask from an address argument corresponding to the single store instruction. (FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3)

Claim 4:

The method of claim 3, wherein the address argument comprises a displacement value and an address value. (FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3)

Claim 6:

The method of claim 1, further comprising the step of computing the mask based upon a data type of the data element. (FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3)

Claim 11:

The method of claim 1, wherein the instruction sequence is without a merge instruction. (Pg. 26, l. 7-10, compare with pg. 21, l. 19 to pg. 22, l. 25).

Claim 12:

A system for aligning and inserting data elements into a first memory in response to an instruction sequence consisting of one or more alignment instructions and a single store instruction, (see FIG. 7, pg. 23-26, FIG. 10, pg. 26-28) *comprising:*

means for receiving a data item that includes a data element to be stored;

means for aligning the data element in a second memory with respect to a predetermined position in the first memory, in response to the one or more alignment instructions;(FIG. 8, pg. 24, l. 1-9, FIG. 11, pg. 27, l. 20 – pg. 28, l. 2)

means for dynamically generating a mask to enable writing of memory bit lines that correspond to the aligned data element, in response to the single store instruction;(FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3) *and*

means for writing the memory bit lines to the first memory under a control of the mask, in response to the single store instruction. (FIG. 9, step 920, pg. 26, l. 3-6, FIG. 12, step 1220, pg. 28, l. 12-15).

Claim 19:

A method for storing data in a memory based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction, comprising the steps of:

aligning the data in a register relative to a location of the data within a target memory address line, in response to the one or more alignment instructions;(FIG. 8, pg. 24, l. 1-9, FIG. 11, pg. 27, l. 20 – pg. 28, l. 2) *and*

storing a portion of the aligned data within the memory under a control of data type information and an address argument specified by the single store instruction, in response to the single store instruction. (FIG. 9, step 920, pg. 26, l. 3-6, FIG. 12, step 1220, pg. 28, l. 12-15).

Claim 20:

The method of claim 19, wherein said storing step stores the portion of the aligned data under the control of a write mask computed from the data type information and the address

specified by the single store instruction. (FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3)

Claim 22:

The method of claim 19, wherein the address argument comprises a displacement value and an address value. (FIG. 9, step 910, pg. 25, l. 1-5, FIG12, step 1210, pg. 28, l. 8-12, FIG. 5, pg. Pg. 17, l. 11 – pg. 19, l. 3)

Claim 24:

The method of claim 19, wherein the instruction sequence is without a merge instruction. (Pg. 26, l. 7-10, compare with pg. 21, l. 19 to pg. 22, l. 25).

Embodiments of the invention can be used in conjunction with other software techniques for accessing big-endian and little-endian data (pg. 28, l. 16-22). For example, the insrtruction sequence of FIG. 7 can be used in conjuction with a first bit swizzling step, such as an `xori` instruction (pg. 29, l. 3-15). In other embodiments, specialized store instructions can be incorporated to implement systems that can access unaligned data. Such an insrruction can perform a store under the control of a mask that selects the bits in a first or second half of an unaligned word, where the first half is those data elements to be stored at an address below an alignment boundary, and the second half is to be stored at an address above an alignment boundary. (Pg. 29, l. 20 – pg. 30, l. 15.) When an aligned data item is stored by the instruction sequence directed towards support for storing unaligned data items, at least one of the specialized store instructions can be a `no-op` (pg. 30, l. 16-18). The alignment boundary can be based on word size, wide word size, memory line size, cache line size, or any other similar natural architectural boundary (pg. 30, l. 20-22).

6. Grounds of Rejection to be Reviewed on Appeal

A. Claims 1-7, 11-13, 17-20, and 22-24 were rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 4,569,016 (Hao, *et al.*)

B. Claims 8-9 and 14-15 were rejected under 35 U.S.C. §103(a) as being obvious to a person having ordinary skill in the art, over U.S. Patent No. 4,569,016 (Hao, *et al.*) in view of Applicant's Admitted Prior Art (AAPA).

C. Claims 10, 16, and 21 were rejected under 35 U.S.C. §103(a) as being obvious to a person having ordinary skill in the art, over U.S. Patent No. 4,569,016 (Hao, *et al.*), in view of U.S. Patent No. 6,167,509 (Sites, *et al.*).

7. **Arguments**

A. The subject matter claimed in each of Claims 1-7, 11-13, 17-20, and 22-24, taken as a whole, are not anticipated by U.S. Patent No. 4,569,016 (Hao, *et al.*)

The invention defined by Applicant's claims 1, 12, and 19 provides for a sequence of one or more alignment instructions and a single store instruction for aligning a data item and storing the data item in memory. Claims 1 and 12 are directed to a method and system for "aligning and inserting data elements into a first memory", and include, *inter alia*, "aligning the data element in a second memory", "dynamically generating a mask to enable writing of memory bit lines . . .", and "writing the memory bit lines to the first memory under a control of the mask, wherein said generating and writing steps are performed in response to the single store instruction". The method is "based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction".

Claim 19 recites method for "storing data in a memory" that includes, *inter alia,* "aligning the data in a register relative to a location of the data within a target memory address line" and "storing a portion of the aligned data within the memory under a control of data type information and an address argument specified by the single store instruction, in response to the single store instruction". The method is "based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction".

The Examiner has relied on U.S. Patent No. 4,569,016 of Hao, *et al.*, for allegedly disclosing the system and method substantially as claimed in Claims 1, 12, and 19. In rejecting claims 1 and 12, the Examiner contends that Hao's rotate and store instructions (Hao, col. 18-19) and Hao's rotate and mask insert instructions (Hao, col. 13) dislose aligning a data element with resepct to a presetermined position in a memory, in resposne to the one or more alignment instructions. The Examiner further contends that Hao's rotate and store instructions (Hao, col.

18, l. 23-24) disclose dynamically generating a mask to enable writing aligned data memory bit lines, and that Hao (col. 18, l. 24-27, col. 13, l. 23-25) dislcose writing the data to memory under control of the single store instruction. In addition, in rejecting claim 19, the Examiner contends that Hao (col. 18, l. 24-27, col. 13, l. 23-25) discloses storing a portion of the aligned data within the memory under a control of data type information and an address argument specified by the single store instruction, in response to the single store instruction.

In order for a reference to anticipate a claim, each and every element set forth in the claim must be found, either expressly or inherently, in the reference. Verdegaal Bros. V. Union Oil Co. of California, 814 F.2d 628,631, 2 USPQ2d 1051,1053 (Fed. Cir. 1987). The reference cited by the Examiner does not disclose, either expressly or inherently, each and every element of Applicant's Claims 1, 6, or 11. Therefore, the rejection under 35 USC 102 should be reversed.

Hao is directed to a system for implementing an instruction set that can perform the functions of rotation, shifting and merging under a mask in a single machine cycle. These functions can be used to perform bit shift, byte alignment, merge and insert operations. One subset of instructions is the rotate with mask instructions. (Col. 12, l. 55 to col. 14, l. 15.) In these instructions, the contents of a register are rotated a specified number of positions, and then are either inserted into another register under control of a mask provided with the instruction, or are ANDed with the mask provided with the instruction and then placed into another register. Another subset of instructions is the rotate and store instructions. (Col. 18, l. 4 to col. 20, l. 17.) In these instructions, the contents of a register are rotated a specified number of positions, and a mask corresponding to the number of rotated bits is generated. The rotated register word is merged with the contents of another register under control of the generated mask, and the merged word is stored in a location pointed to by the contents of a third register. In some of the rotate and store instructions, an offset for the location storage is built into the instruction, and the value of the third register is incremented by the offset. In other rotate and store instructions, extra bytes are included with the stored word to indicate which bytes were altered by the merge.

1. Claims 1 and 12 are not anticipated by Hao.

The Examiner cites two passages in Hao in rejecting independent claims 1 and 12. One of these passages discloses the rotate with mask instructions, the other the rotate and store instructions. The combination of these two instructions does not fully disclose the elements of claim 1 and 12. The rotate with mask instructions disclose rotating a data word a specified number of bits and then inserting the rotated data word from one register into another register under control of a mask provided with the instruction, and is known prior art. The number of bits to be rotated is included with the instruction. However, Hao does not disclose anything regarding how the number of bits to be rotated is determined, and thus Hao does not disclose *aligning a data element . . . with respect to a predetermined position in . . . memory*, as essentially recited in claims 1 and 12. Furthemore, Hao discloses inserting the rotated data item into another register under control of the mask provided, or ANDing it with the mask and placing it into another register. Hao does not disclose *writing* the data item *to . . . memory under a control of the mask*, as essentially recited in claims 1 and 12. Note that the mask in the rotate with mask instruction is provided with the instruction, and is not dynamically generated as recited in claims 1 and 12.

Similarly, Hao's rotate and store instructions rotate the contents of a first register and merge them into a second register under control of a mask generated during the instruction, before writing the data from the second register into storage. Again, this is different from Applicant's instruction sequence, in which *the data element [is aligned] in a second memory*, and then *writing the memory bit lines to the first memory under a control of the mask*, as essentially recited in claims 1 and 12. First, Hao's mask is generated from the rotation, whereas Applicant's mask is generated in response to a store instruction, and is not generated as part of the alignment instruction. Furthermore, Hao's mask is used to merge data from one register into another register, not to control the writing of data from the second register into memory. In addition, as with the rotate with mask instruction, there is no disclosure in Hao as to whether to the number of rotated positions is based on *aligning the data element. . . with respect to a predetermined position in . . . memory*, as recited in claims 1 and 12. Although Hao disloses that one purpose of these instruction is to align byte data, Hao nowhere discloses that the number of

bits the register contents are rotated is is based on *aligning the data element. . . with respect to a predetermined position in . . . memory.* Thus, neither of Hao's rotate with mask instructions nor rotate and store instructions disclose *aligning the data . . . relative to a location of the data within a target memory address line*, or *writing the memory bit lines to the first memory under a control of the mask*, as recited in claims 1 and 12.

With regard to the arguments presented by the Examiner in paragraph 18 of the Final Office Action, it is the Examiner's arguments of paragraph 3 of the Action that are conclusory. Claims 1 and 12 recite *aligning the data in a register relative to a location of the data within a target memory address line*, and the Examiner asserts that this limitation is taught by the rotate with mask instruction and by the rotate and store instruction. Hao does not teach or suggest how the number of bit positions to be rotated is determined, and the Examiner has not shown otherwise.

Further, the Examiner misconstrues and confuses Applicant's arguments in the Resposne filed on November 2, 2004. The Examiner begins by repeating Applicant's argument that Hao does not disclose *aligning the data element in a second memory with respect to a predetermined position in the first memory*, states that this claim recitation was presumed to encompass an implementation described in Applicant's specification at page 23, and then concludes that the Applicant did not mean this claim recitation to encompass the cited implementation based on Applicant's argument that Hao does not disclose this limitation. The implementation described in Applicant's specification from pgs. 23-26 uses an alignment network to align and store data. The fact that Hao does not disclose such a network does not suggest otherwise.

Since Hao does not disclose *aligning the data . . . relative to a location of the data within a target memory address line*, or *writing the memory bit lines to the first memory under a control of the mask*, as set forth in Applicant's claims 1 and 12, Hao does not anticipate these claims, and the rejection of these claims should be reversed.

2. Claim 19 is not anticipated by Hao.

The Examiner relies on Hao's rotate with mask instructions and rotate and store instruction in rejecting claim 19. Although both instructions disclose rotating a data item in a register by a specified number of bits, neither discloses *aligning the data in a register relative to a location of the data within a target memory address line,* as recited in claim 19. Further, Hao discloses storing the full, merged word into main storage, not *storing a portion of the aligned data within the memory*, as recited in claim 19. The Examiner asserts that Hao's rotate with mask instruction disclose storing under the control of immediate data, and that this discloses storing data under the control of data type information. The Examiner is incorrect in this assertion, however. Hao does not include *data type information,* as recited in claim 19, with the rotate with mask instruction, for immediate data does not constitue data type information, and the rotate with mask instruction does not include writing data to memory. Thus, neither of Hao's rotate with mask instructions nor rotate and store instructions disclose *aligning the data in a register relative to a location of the data within a target memory address line*, or *storing a portion of the aligned data within the memory under a control of data type information*, as recited in claim 19.

In addition, the Examiner's argument in paragraph 19 of the Final Office Action is disingenuous. Storing a full word into memory does not disclose storing a portion of a word into memory when the purpose is to store only a portion of a word. Further, there is no positive recitation of a merge in claim 19. Hao only discloses storing a full word into memory, not *storing a portion of a word under control of data type information.*

Since Hao does not disclose *aligning the data in a register relative to a location of the data within a target memory address line*, or *storing a portion of the aligned data within the memory under a control of data type information*, as claimed in claim 19, Hao does not anticipate these claims, and the rejection of these claims should be reversed.

3. Claim 3 is not anticipated by Hao.

Dependent claim 3 recites *computing the mask from an address argument.* Although the rotate and store instruction does disclose an address argument, the mask is not computed from the address but rather is generated from the rotation. The Examiner cites col. 13, l. 16-30 as disclosing a mask being taken from an immediate address. The cited passage discloses a Rotate Immediate then Mask Insert instruction. Although the cited text describing this instruction includes the phrase "generated mask", there is no indication as to how this mask is generated. There is no disclosure in that passage that the mask is taken from an immediate address. Further, this phrase contradicts the general description of the Rotate with Mask instructions that the mask is provided with the instruction, and the register schematic shows a mask starting at bit 21. Thus Applicant interprets this passage as meaning that the mask is provided with the instruction, and is not computed from an address argument.

The Examiner also cites col. 18, l. 42-55 as disclosing a mask being dynamically generated from an address. This passage discloses 4-bit byte marks, whose pattern of ones and zeros are indicative of which bytes of the stored word were altered by the merge. Thus, these byte marks are not used as a mask but rather to keep track of which bytes were altered as a result of the rotate and merge. The pattern of the byte marks is consistent the pattern of byte alteration that would result from a rotation being performed before a merge. Further, Hao states that the merged word is stored with the byte marks, implying that the byte marks are also stored in memory. Hao does not state that the byte marks are used as a mask to control the writing of the merged word into memory. Thus, since the byte marks do not constitute a mask, this passage of Hao does not disclose *computing the mask from an address argument*, as essentially claimed in dependent claim 3.

4. Claims 4 and 22 are not anticipated by Hao.

Dependent claims 4 and 22 recite that *the address argument comprises a displacement value and an address value.* The Examiner cites Hao, col. 18, l. 26-28 as disclosing a displacement value and an address value. However, although the cited passage discloses an address argument, this address does not include displacement value as part of the argument, but

is rather an auto-pre-increment that is built into the particular instruction. Note that all of Hao's rotate and store instructions include an address argument, but in some of those instructions, specifically the rotate and store with update instructions, the merged word is stored at a fixed increment from the value of the address argument where the fixed increment (+4) is built into the instruction, and is not an argument to the instruction. Thus, Hao does not disclose an *address argument* [that] *comprises a displacement value and an address value*, as essentially recited in claims 4 and 22.

5. Claims 6 and 20 are not anticipated by Hao.

Dependent claims 6 and 20 both essentially recite *computing the mask based upon a data type of the data element.* As stated above in connection with claim 19, Hao does not disclose data type information as part of the rotate with mask instruction, and thus Hao does not anticipate claims 6 or 20.

6. Claims 11 and 24 are not anticipated by Hao.

Dependent claims 11 and 24 recite that *the instruction sequence is without a merge instruction.* As discussed above, that Examiner argues that because Hao's instructions are executed as single instructions, there do not include a merge insrtruction, even though Hao's disclosure explicitly states that a merge occurs in each of those instructions. (Col. 13, l. 1-5, col. 18, l. 15-30.) This argument is disingenuous, for the presence of a merge within an instruction does not anticipate the explicitly claimed lack of a merge within an instruction sequence. Thus, Hao does not disclose that *the instruction sequence is without a merge instruction.*

The Examiner's argument in paragraph 20 of the Final Office Action is also disingenuous. As opposed to the Examiner's argument in paragraph 19, in which was argued that storing a full word discloses storing a portion of a word, it is now argued that because Hao's rotate and store instructions are single instructions, these instructions disclose a lack of a merge instruction even though both instructions incorporate a merge. Applicant urges that the presence of a merge within the rotate and store instruction discloses a merge instruction, contrary to that recited in claims 11 and 24.

7. Claims 2, 5, 7, 13, 17-18, and 23 are not anticipated by Hao.

Claims 2, 5, and 7 depend from claim 1 and claims 13 and 17-18 depend from claim 12, and therefore include the elements claimed in Claims 1 and 12, respectively. Claims 2, 5, 7, 13, and 17-18 are therefore believed patentably distiguished and not anticipated by Hao, for the reasons given aove. Claim 23 depends from claim 19, and therefore include the elements claimed in Claims 19. Claim 23 is therefore believed patentably distiguished and not anticipated by Hao, for the reasons given aove.

B. The subject matter claimed in each of Claims 8-9 and 14-15, taken as a whole, are not obvious to a person having ordinary skill in the art, over U.S. Patent No. 4,569,016 (Hao, *et al.*) in view of Applicant's Admitted Prior Art (AAPA).

Dependent claims 8 and 14 recite *computing and checking parity information corresponding to the data element*, while dependent claims 9 and 15 recite *computing and checking error correction code (ECC) information corresponding to the data element.*

In rejecting claims 8-9 and 14-15, the Examiner has stated that it would be obvious to one of ordinary skill in the art to include logic for computing and checking parity information and error correction codes. Claims 8-9 depend upon claim 1, and claims 14-15 depend upon claim 12.

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the reference itself or in the knowledge generally available to one of ordinary skill in the art, to modify the reference teaching. In re Vaeck, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991). Second, there must be a reasonable expectation of success. In re Merck & Co., Inc., 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). Finally, the prior art reference must teach or suggest all the claim limitations. In re Royka, 490 F.2d 981, 180 USPQ 580 (CCPA 1974). The teaching or suggestion to make the claimed invention and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. If an independent claim is non-obvious under section 103, then any claim depending therefrom is non-obvious. In re Fine, 837 F.2d 1071, 5 USPQ2d 1596 (Fed.

Cir. 1988). The reference cited by the Examiner is legally deficient to establish a *prima facie* case of obviousness against independent claims 1, 6 and 11. Therefore, the rejection under 35 USC 103 of dependent claims 3-5, 8-10, 13, and 15-20 should be reversed.

As stated above, Hao does not teach or suggest all of the claimed features of independent claims 1, 12, and 19, nor are these features suggested by Applicant's admitted prior art (AAPA). Specifically, Hao does not teach or suggest *aligning the data . . . relative to a location of the data within a target memory address line*, or *writing the memory bit lines to the first memory under a control of the mask*, as recited in claims 1 and 12, or *aligning the data in a register relative to a location of the data within a target memory address line*, or *storing a portion of the aligned data within the memory under a control of data type information*, as recited in claim 19.

Therefore, the combination of Hao and AAPA fails to teach or suggest all the claim limitations of dependent claims 8-9 and 14-15. Accordingly, claim 1 and its dependent claims 8-9, and claim 12 and its dependent claims 14-15, are not rendered obvious over the combination of Hao and AAPA. The Examiner has failed to establish a *prima facie* case of obviousness. Therefore, the rejection of claims 1 and 12 and their respective dependent claims 8-9 and 14-15 should be reversed.

C. The subject matter claimed in each of Claims 10, 16, and 21, taken as a whole, are not obvious to a person having ordinary skill in the art, over U.S. Patent No. 4,569,016 (Hao, *et al.*), in view of U.S. Patent No. 6,167,509 (Sites, *et al.*).

Dependent claims 10, 16, and 21 essentially recite *intermediately storing the memory bit lines from the second memory to a read-write buffer before said writing step.* Claim 10 depends from claim 1, claims 16 depends from claim 12, and claim 21 depends from claim 19. The Examiner concedes that Hao does not disclose the use of a read-write buffer, and then cites Sites as disclosing storing memory bit lines to a read-write buffer before writing.

However, as stated above, Hao does not teach or suggest all of the claimed features of independent claims 1, 12, and 19, nor are these features suggested by Sites. Sites is directed to a high performance RISC type of CPU and features branch prediction and instruction prefetching

based on the branch prediction to improve performance. Sites does not rectify the above mentioned deficiencies of Hao, and does not teach or suggest *aligning the data . . . relative to a location of the data within a target memory address line*, or *writing the memory bit lines to the first memory under a control of the mask*, as recited in claims 1 and 12, or *aligning the data in a register relative to a location of the data within a target memory address line*, or *storing a portion of the aligned data within the memory under a control of data type information*, as recited in claim 19.

Therefore, the combination of Hao and Sites fails to teach or suggest all the claim limitations of dependent claims 10, 16, and 21. Accordingly, claim 1 and its dependent claim 10, claim 12 and its dependent claim 16, and claim 19 and its dependent claim 21 are not rendered obvious over the combination of Hao and Sites. The Examiner has failed to establish a *prima facie* case of obviousness. Therefore, the rejection of claims 1, 12 and 19 and their respective dependent claims 10, 16, and 21 should be reversed.

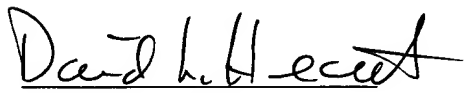
8. Conclusion

The Examiner has contended that the combination of Hao's rotate with mask instruction and rotate and store instruction anticipates the instruction step sequences claimed in Applicants independent claims 1, 12, and 19. Although Hao's instructions disclose rotating data items in a register, merging data items from one register into another under control of a mask, and writing the register contents to storage, Hao does not disclose, teach or suggest aligning a data element with respect to a predetermined position in memory or writing either the memory bit lines corresponding to the aligned data or a portion thereof to memory under control of a mask, as essentially recited in claims 1, 12, and 19. Furthermore, Hao does not disclose computing a mask from an address argument or the data type of the data element, as essentially reicted in dependent claims 3, 6, and 20, and Hao does not dislcose an instruction sequence that does not include a merge instruction, as reicted in claims 11 ansd 24.

Therefore, the Examiner has failed to show that Claims 1-7, 11-13, 17-20, and 22-24of the application are anitcipated. Further, the Examiner has failed to establish a *prima facie* case of

obviousness of Claims 8-10, 14-16, and 21 of the application. Reversal of the rejections and allowance of the rejected claims is urged.

Respectfully submitted,

By: 
David L. Heath
Reg. No. 46,763

Mailing Address:

F. Chau & Associates, LLP

130 Woodbury Road
Woodbury, NY 11797
(516) 692-8888
(516) 692-8889 (FAX)

CLAIMS APPENDIX

1. A method for aligning and inserting data elements into a first memory based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction, comprising the steps of:

given a data item that includes a data element to be stored,

aligning the data element in a second memory with respect to a predetermined position in the first memory, in response to the one or more alignment instructions;

dynamically generating a mask to enable writing of memory bit lines that correspond to the aligned data element; and

writing the memory bit lines to the first memory under a control of the mask, wherein said generating and writing steps are performed in response to the single store instruction.

2. The method of claim 1, wherein the second memory is a register.

3. The method of claim 1, further comprising the step of computing the mask from an address argument corresponding to the single store instruction.

4. The method of claim 3, wherein the address argument comprises a displacement value and an address value.

5. The method of claim 4, wherein the address value specifies a particular register.

6. The method of claim 1, further comprising the step of computing the mask based upon a data type of the data element.

7. The method of claim 1, wherein the predetermined position in the first memory corresponds to a target position within a memory line.

8. The method of claim 1, further comprising the step of computing and checking parity information corresponding to the data element.

9. The method of claim 1, further comprising the step of computing and checking error correction code (ECC) information corresponding to the data element.

10. The method of claim 1, further comprising the step of intermediately storing the memory bit lines from the second memory to a read-write buffer before said writing step.

11. The method of claim 1, wherein the instruction sequence is without a merge instruction.

12. A system for aligning and inserting data elements into a first memory in response to an instruction sequence consisting of one or more alignment instructions and a single store instruction, comprising:

means for receiving a data item that includes a data element to be stored;

means for aligning the data element in a second memory with respect to a predetermined position in the first memory, in response to the one or more alignment instructions;

means for dynamically generating a mask to enable writing of memory bit lines that correspond to the aligned data element, in response to the single store instruction; and

means for writing the memory bit lines to the first memory under a control of the mask, in response to the single store instruction.

13. The system of claim 12, wherein said system exploits partial line write capabilities of the first memory.

14. The system of claim 12, further comprising logic for computing and checking parity information corresponding to the data element.

15. The system of claim 12, further comprising logic for computing and checking error correction code (ECC) information corresponding to the data element.

16. The system of claim 12, further comprising:

a CPU;

a read-write buffer for intermediately storing, under a control of the CPU, the memory bit lines from the second memory before said writing step.

17. The system of claim 12, wherein the first memory comprises a cache, and said means for writing writes the data element to the cache under the control of the mask.

18. The system of claim 12, wherein the data item is a data word.

19. A method for storing data in a memory based upon an instruction sequence consisting of one or more alignment instructions and a single store instruction, comprising the steps of:

aligning the data in a register relative to a location of the data within a target memory address line, in response to the one or more alignment instructions; and

storing a portion of the aligned data within the memory under a control of data type information and an address argument specified by the single store instruction, in response to the single store instruction.

20. The method of claim 19, wherein said storing step stores the portion of the aligned data under the control of a write mask computed from the data type information and the address specified by the single store instruction.

21. The method of claim 19, further comprising the step of intermediately storing the aligned data from the register to a read-write buffer before said storing step.

22. The method of claim 19, wherein the address argument comprises a displacement value and an address value.

23. The method of claim 22, wherein the address value specifies a particular register.

24. The method of claim 19, wherein the instruction sequence is without a merge instruction

25-29. (Canceled)

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.

PTO/SB/17 (12-04)
Approved for use through 07/31/2006. OMB 0651-0032
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

Effective on 12/08/2004.
Fees pursuant to the Consolidated Appropriations Act, 2005 (H.R. 4818).

FEE TRANSMITTAL For FY 2005

☐ Applicant claims small entity status. See 37 CFR 1.27

TOTAL AMOUNT OF PAYMENT ($) 500.00

Complete if Known	
Application Number	09/940,911
Filing Date	August 28, 2001
First Named Inventor	Gschwind, Michael K.
Examiner Name	Choi, Woo H.
Art Unit	2189
Attorney Docket No.	YOR920010606 (8728-545)

METHOD OF PAYMENT (check all that apply)

☐ Check ☐ Credit Card ☐ Money Order ☐ None ☐ Other (please identify): _____

☒ Deposit Account Deposit Account Number: 50-0150 Deposit Account Name: IBM/YORKTOWN HEIGHTS

For the above-identified deposit account, the Director is hereby authorized to: (check all that apply)

☒ Charge fee(s) indicated below ☐ Charge fee(s) indicated below, **except for the filing fee**

☒ Charge any additional fee(s) or underpayments of fee(s) under 37 CFR 1.16 and 1.17 ☐ Credit any overpayments

WARNING: Information on this form may become public. Credit card information should not be included on this form. Provide credit card information and authorization on PTO-2038.

FEE CALCULATION

1. BASIC FILING, SEARCH, AND EXAMINATION FEES

	FILING FEES		SEARCH FEES		EXAMINATION FEES		
Application Type	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fee ($)**	**Small Entity Fee ($)**	**Fees Paid ($)**
Utility	300	150	500	250	200	100	
Design	200	100	100	50	130	65	
Plant	200	100	300	150	160	80	
Reissue	300	150	500	250	600	300	
Provisional	200	100	0	0	0	0	

2. EXCESS CLAIM FEES

Fee Description	**Fee ($)**	**Small Entity Fee ($)**
Each claim over 20 or, for Reissues, each claim over 20 and more than in the original patent	50	25
Each independent claim over 3 or, for Reissues, each independent claim more than in the original patent	200	100
Multiple dependent claims	360	180

Total Claims		**Extra Claims**	**Fee ($)**		**Fee Paid ($)**
_____	- 20 or HP =	_____	x _____	=	_____

HP = highest number of total claims paid for, if greater than 20

Indep. Claims		**Extra Claims**	**Fee ($)**		**Fee Paid ($)**
_____	- 3 or HP =	_____	x _____	=	_____

HP = highest number of independent claims paid for, if greater than 3

Multiple Dependent Claims

Fee ($)	**Fee Paid ($)**
_____	_____

3. APPLICATION SIZE FEE

If the specification and drawings exceed 100 sheets of paper, the application size fee due is $250 ($125 for small entity) for each additional 50 sheets or fraction thereof. See 35 U.S.C. 41(a)(1)(G) and 37 CFR 1.16(s).

Total Sheets		**Extra Sheets**		**Number of each additional 50 or fraction thereof**	**Fee ($)**		**Fee Paid ($)**
_____	- 100 =	_____	/ 50 =	_____ (round up to a whole number)	x _____	=	_____

4. OTHER FEE(S)

	Fees Paid ($)
Non-English Specification, $130 fee (no small entity discount)	
Other: Appeal Brief	500.00

SUBMITTED BY

Signature	David L. Heath	Registration No. (Attorney/Agent) 46,763	Telephone 516-692-8888
Name (Print/Type)	David L. Heath		Date July 25, 2005